Санкт-Петербургский государственный университет

Кафедра системного программирования

Ярош Дмитрий Сергеевич

Мобильное приложение для изучения биологии

Дипломная работа

Научный руководитель:
к.ф.-м.н., доцент К. Ю. Романовский

Консультант:
вед. прогр. ООО «КиберТех Лабс» А. В. Корнилова

Рецензент:
глава мобильной разработки ООО «Мел Саенс» П. М. Катунин

Санкт-Петербург
2021

SAINT-PETERSBURG STATE UNIVERSITY

Software Engineering

Iarosh Dmitrii

Mobile application for studying biology

Graduation Thesis

Scientific supervisor:
assistant professor Konstantin Romanovsky

Consultant:
lead Programmer at CyberTech Co., Ltd Anastasia Kornilova

Reviewer:
head of mobile development at MEL Science Pavel Katunin

Saint-Petersburg
2021

Оглавление

Введение

Одним из эффективных способов изучения естественных наук является постановка экспериментов и исследование природных явлений. В биологии для изучения окружающего мира можно использовать микроскопию, которая позволяет проводить различные эксперименты и исследовать образцы с помощью сильного увеличения. Существуют разные виды микроскопов, отличающиеся увеличительными способностями и способами формирования изображения образца. В последние годы стоимость оптических микроскопов снизилась, что позволяет приобрести свой собственный микроскоп и использовать его в домашней лаборатории для изучения биологии на практике. Серьезным препятствием на пути использования данного прибора в домашних условиях является отсутствие возможности получить цифровое изображение образца и сохранить его для дальнейшего исследования. Кроме того оптические микроскопы имеют небольшой угол обзора и маленькую глубину резкости — это не позволяет рассмотреть образец целиком без постоянного корректирования положения приборного столика вручную.

Возможным способом решения всех вышеперечисленных проблем является захват видео с микроскопа и его цифровая обработка с целью получения полного сфокусированного изображения образца. К сожалению, большинство существующих оптических микроскопов не предназначены для захвата с них видеоданных. В связи с этим компания MELScience [16], занимающаяся образовательными технологиями в области естественных наук, разработала мобильный микроскоп — устройство, подготовленное для захвата видеоданных образца смартфоном. Помимо этого, данное устройство обладает дистанционно управляемым приборным столиком, способным перемещаться как в горизонтальной плоскости, так и в вертикальной для обеспечения обзора всех частей образца на различном удалении от оптики микроскопа. Вместе со смартфоном мобильный микроскоп образует систему, позволяющую изучать биологические образцы с применением технологий цифровой обработки изображений в домашних условиях.

Мобильный микроскоп оснащен платой Arduino для управления приборным столиком, освещением и связи с мобильным приложением. Необходимо реализовать программное обеспечение для данной платы, которое позволит дистанционно управлять микроскопом и получать сообщения об ошибках с помощью специального протокола, который также предстоит разработать.

Для взаимодействия с мобильным микроскопом и изучения биологии необходимо реализовать мобильное приложение. Одной из задач мобильного приложения является обработка изображений образцов, использующая такие подходы, как:

1. фокус-стекинг (focus-stacking) — получение сфокусированного изображения образца путем объединения нескольких изображений, содержащих в фокусе лишь его часть;

2. ститчинг (stitching) — объединение нескольких частично перекрывающихся изображений частей образца в одно большое изображение, содержащее весь образец целиком;

3. 3d-реконструкция (3D reconstruction) — воссоздание 3d-модели образца путем объединения набора кадров, сделанных при разном удалении образца от линзы микроскопа.

Также необходимо предоставить пользователю возможность знакомиться с дополнительными материалами об изучаемом образце в приложении и проходить различные тесты и лабораторные работы для закрепления знаний. В проверочных работах могут использоваться полученные пользователем и обработанные с помощью данного приложения изображения образцов. Для этого мобильное приложение должно поддерживать возможность редактирования изображений, чтобы пользователь мог отметить на изображении определённые области, подписать обнаруженные объекты и их части.

В рамках данной работы будет описана реализация программного обеспечения для данной системы — мобильное приложение для ОС Android [12] и ПО для встроенного контроллера микроскопа.

1. Постановка задачи

Цель работы — разработать ПО для мобильного микроскопа, а именно: программное обеспечение для контроллера микроскопа, позволяющее дистанционно управлять им, и мобильное приложение для ОС Android. Мобильное приложение должно иметь следующий набор функций:

- установление связи с микроскопом;
- управление микроскопом (передвижение предметного столика, включение подсветки образца);
- захват видеоданных с микроскопа;
- цифровая обработка захваченных видеоданных с использованием мобильной библиотеки, разработанной в ходе дипломной работы студента кафедры системного программирования Владимира Кутуева [23];
- встроенный фоторедактор для выполнения различных заданий с использованием изображений образцов.

Для достижения цели были поставлены следующие задачи:

1. сделать обзор технологий для разработки мобильного приложения для ОС Android;
2. спроектировать и реализовать ПО для управления микроскопом на базе Arduino [4];
3. разработать протокол обмена сообщениями для мобильного приложения и микроскопа;
4. спроектировать и реализовать мобильное приложение для ОС Android, поддерживающее вышеперечисленные функции;
5. внедрить в разработанное приложение библиотеку для обработки изображений с микроскопа.

2. Обзор

В данном разделе представлены основные средства и инструменты, используемые для реализации ПО для микроскопа и клиент-серверного мобильного приложения, а также описаны используемые алгоритмы для обработки изображений и их требования к исходным кадрам.

2.1. Микроскоп

Микроскоп представляет из себя пластиковое основание с креплением для смартфона и линзой, на которой формируется изображение образца. Изображение данного прибора можно увидеть на рис. 1. Сверху к основанию на подвижной установке прикреплен приборный столик, на который помещается образец для исследования. Для передвижения приборного столика используются шаговые электромоторы NEMA 17 [1], управляемые с помощью платы, расположенной в основании микроскопа. Для подсветки образца используется источник света, закрепленный над приборным столиком. Данный осветительный прибор может быть включен и выключен с помощью команд микроконтроллера. В качестве платы управления в микроскопе используется плата Ардуино Уно (Arduino Uno) [5]. Данное устройство представляет из себя микроконтроллер с вычислительной мощностью 16 МГц, 32 Кб флеш-памяти и 14 портами для подключения устройств. Для связи со смартфоном к плате дополнительно подключен внешний Bluetooth-модуль DSD Tech HC-05 [20]. Задачей микроконтроллера является управление приборным столиком — смещение его в плоскости XY и по вертикали (с возможностью одновременного движения по нескольким осям), управление освещением образца, а также поддержание связи с мобильным приложением через модуль Bluetooth и прием команд на управление микроскопом. Дополнительно для каждой оси предусмотрены ограничители, позволяющие приборному столику двигаться только в пределах установленного при изготовлении микроскопа интервала. В случае попытки выдвинуть приборный столик за пределы данного интервала на контроллер посылается специальный сигнал, который должен быть

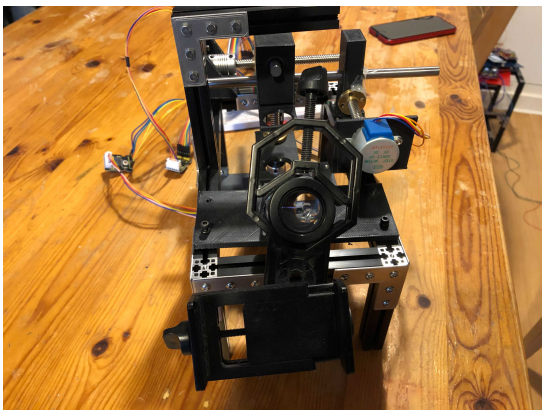

Рис. 1: Микроскоп

обработан для предотвращения дальнейших попыток выдвижения столика в запрещенную зону.

2.2. Средства мобильной разработки

Существуют различные подходы к разработке мобильных приложений — нативные и с использованием кроссплатформенных фреймворков. Среди нативных подходов можно выделить реализацию приложений на низком уровне с помощью языка C++ и более высокоуровневую разработку с использованием Kotlin для ОС Android и Swift для iOS [3]. Данные способы разработки мобильных приложений позволяют легко получать доступ к аппаратному обеспечению телефона, а также использовать все возможности соответствующей платформы. Дополнительным преимуществом является поддержка нативных методов разработки приложений компаниями-производителями операционных систем для смартфонов — это позволяет использовать все новшества ОС непосредственно сразу после ее обновления. Также важно учесть, что уже существующие приложения компании MELScience реализованы без использования фреймворков, таким образом при нативном подходе появляется возможность переиспользовать часть уже существующего кода. Подход с использованием кроссплатформенного фреймворка подразумевает использование дополнительной прослойки, позволяющей использовать одну кодовую базу для обоих платформ. Среди таких фреймворков стоит выделить следующие.

- **React Native** [10] — кроссплатформенный фреймворк, позволяющий реализовывать мобильные приложения с использованием языка программирования Javascript. Написанный на нем код интерпретируется с помощью metro bundler [9], а затем передаётся нативному сборщику соответствующей ОС. Одними из главных преимуществ данного решения является возможность переиспользования опыта веб-разработки в мобильных приложениях и обновления приложения, происходящие напрямую, в обход Google Play [15] и AppStore [2]. Среди недостатков стоит выделить сложности при интеграции с библиотеками, реализованными с использованием нативного подхода, возможные проблемы при отладке программ — низкоинформативные сообщения об ошибках, наличие различных интерпретаторов для сборки debug и release версий приложения, что может привести к неожиданному появлению ошибок при использовании приложения клиентами, а так же отсутствие возможности реализации произвольного пользовательского интерфейс без использования средств нативной разработки.

- **Xamarin** [19] — кроссплатформенный фреймворк от Microsoft [17], позволяющий реализовывать мобильные приложения на языке C#. Данное решение позволяет использовать библиотеки, предназначенные для платформы .Net [18] в мобильной разработке, однако предоставляет кроссплатформенность только бизнес-логики и работы с сетью, в то время как пользовательский интерфейс реализуется с помощью нативных средств разработки. Недостатками Xamarin являются размер скомпилированного приложения по сравнению с нативной версией, отсутствие удобной специализированной интегрированной среды разработки, а также необходимость приобретения данного фреймоворка (около 1000$ в год).

- **Flutter** [14] — активно развивающийся кроссплатформенный фреймворк, использующий отличающийся от предыдущих подход к разработке и специальный язык Dart. В данном решении пользовательский интерфейс реализуется самостоятельно, а затем отрисо-

вывается фреймворком с использованием низкоуровненого интерфейса платформы. В отличие от остальных фреймворков Flutter взаимодействует с ОС устройства на низком уровне, тем самым увеличивая скорость работы до нативной. Среди недостатков данного фреймворка стоит выделить малое количество существующих библиотек, большой объем скомпилированного файла.

- **Qt** [7] — крупный фреймворк для кроссплатформенной разработки на C++, который позволяет разрабатывать приложения и для мобильных платформ. К сожалению, весь пользовательский интерфейс при использовании данного фреймоврка необходимо реализовывать самому, так как данное решение не имеет встроенного способа создания пользовательского интерфейса для ОС Android и iOS.

Несмотря на возможность использовать один код для обеих платформ все вышеперечисленные решения предполагают наличие части приложения, разработанной с помощью нативного метода — для взаимодействия с аппаратным обеспечением телефона. Так как основной задачей приложения является работа с камерой смартфона, обработка изображений и их редактирование, было принято решение использовать нативный подход для разработки приложения с использование языка программирования Kotlin. Дополнительным аргументом послужила возможность переиспользовать код существующих нативных приложений компании MELScience.

2.3. Алгоритмы обработки изображений

После захвата изображения с микроскопа, его необходимо обработать для дальнейшего изучения. Для обработки изображений в данной работе будет использоваться кроссплатформенная мобильная библиотека, разработанная Владимиром Кутуевым в ходе дипломной работы на кафедре системного программирования [23].

2.3.1. Фокус-стекинг

Оптические микроскопы обладают небольшой глубиной резкости, из-за чего большинство образцов не могут находиться в фокусе целиком. Для получения полностью сфокусированного изображения можно использовать алгоритмы фокус-стекинга — метода, который объединяет несколько изображений, сделанных с разными фокусными расстояниями. В результате получается изображение с большей глубиной резкости, чем у любого из отдельных исходных изображений (рис. 2). В мобильной библиотеке, которая будет использоваться приложением для обработки изображений, реализовано 4 различных алгоритма фокус-стекинга.

- Алгоритм, основанный на пикселях (pixel-based) [11] — для каждого пикселя исходных изображений определяется уровень его сфокусированности, и самые сфокусированные пиксели формируют итоговое изображение

- Алгоритм, основанный на соседях пикселя (neighbour-based) [11] — для определения сфокусированности пикселя используется не только его значение, но и значение соседних пикселей. Это аргументировано тем, что области находящиеся в фокусе всегда состоят из большого числа пикселей, таким образом сфокусированные пиксели всегда сгруппированы вместе.

- Алгоритм, основанный на вещественных вейвлет преобразованиях (real wavelet) [11] — в отличие от предыдущих алгоритмов, данный подход анализирует изображение целиком и получает сфокусированный результат путем прямых и обратных вейвлет преобразований.

- Алгоритм, основанный на комплексных вейвлет преобразованиях (complex wavelet) [8] — повышает эффективность алгоритма, использующего целочисленные вейвлет преобразования, путем добавления комплексной составляющей.

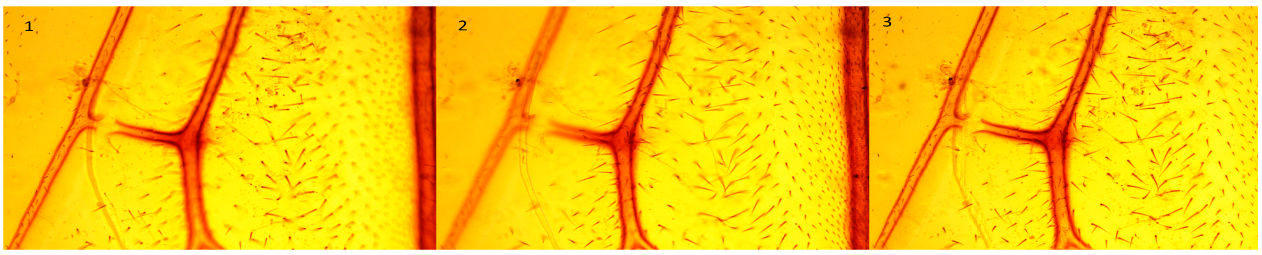

Рис. 2: Пример фокус-стекинга: первые два изображения — исходные, третье — результат фокус-стекинга

Данные алгоритмы отличаются качеством итогового изображения, а также количеством времени, затраченным на его получение. Например, алгоритм, основанный на пикселях, выдает изображение невысокого качества, однако способен работать в режиме реального времени. Данный алгоритм может быть использован для демонстрации пользователю предварительного результата сразу после окончания съемки. Пока пользователь рассматривает полученное таким образом изображение образца, медленный, но строящий более качественное и детализированное изображение алгоритм, основанный на вейвлет преобразованиях, обрабатывает записанные кадры, а затем обновляет результат для пользователя. В качестве входа каждый из описанных алгоритмов принимает видео, на котором образец плавно перемещается на приборном столике по вертикальной оси (возможна также обработка в реальном времени — пошаговая работа алгоритмов для каждого нового кадра, записанного смартфоном). Благодаря встроенной в библиотеку фильтрации кадров из записанного видео извлекаются только кадры, содержащие разные части образца в фокусе, затем они передаются алгоритмам фокус-стекинга для получения полностью сфокусированного изображения.

2.3.2. Автофокус

Несмотря на то, что существует возможность получить изображение, на котором все части образца находятся в фокусе, пользователи часто хотят изучать образцы самостоятельно, вручную меняя высоту приборного столика. Устройство микроскопа таково, что интервал высоты

приборного столика, в котором хотя бы часть образца находится в фокусе, невелик по сравнению с множеством всех возможных положений по высоте. В связи с этим необходимо поддерживать возможность автофокуса — поиска положения приборного столика, в котором хотя одна часть образца находится в фокусе. Для решения данной задачи в библиотеке реализованы фокусные меры. Фокусная мера — функция, принимающая на вход изображение и оценивающая его сфокусированность как некоторое число. Алгоритм автофокуса подразумевает проход по всем положениям приборного столика с обработкой кадров с помощью фокусной меры. Положение, при котором фокусная мера максимальна, считается самым сфокусированным и гарантированно содержит часть образца в фокусе. В мобильной библиотеке реализованы следующие фокусные меры (наиболее эффективные для применения в микроскопии [6]):

- Laplacian Variance;
- Tenengrad operator;
- Vollath F4;
- modified Laplacian.

Таким образом, алгоритм автофокуса принимает на вход последовательность кадров, на которых изображен образец при проходе приборным столиком всего доступного интервала для его перемещения по высоте. Обрабатывая кадры в реальном времени, алгоритм выдает необходимую позицию микроскопа сразу после окончания движения приборного столика.

3. Программное обеспечение для микроскопа

Для управления приборным столиком и подсветкой в микроскопе установлен контроллер Arduino Uno, подключенный к плате Bluetooth DSD TECH HC-05 для поддержания связи с мобильным приложением. Для работы данного контроллера было необходимо реализовать программное обеспечение. Для разработки был выбран язык Си, т.к. он является основным языком разработки для платформы Arduino. Главными требованиями к данному ПО были:

- возможность управления приборным столиком и освещением с помощью команд, передаваемых по Bluetooth;
- параллельное исполнение нескольких команд (например, движение столика по нескольким осям одновременно) с возможностью отмены как определённой команды, так и всех команд одновременно;
- уведомление о завершении выполнения команды;
- информирование о возможных ошибках, связанных с достижением приборным столиком границы интервала, в котором разрешено его перемещение, а так же об ошибках соединения и распознавания команд;
- хранение информации о текущем положении приборного столика и возможность запросить данную информацию.

3.1. Архитектура

Для обеспечения постоянного приема новых команд, а также для поддержания псевдопараллельности исполнения существующих был использован шаблон проектирования «Команда», который позволяет сохранить информацию о действии, которое необходимо выполнить в виде структуры. Принцип работы разработанного решения представлен

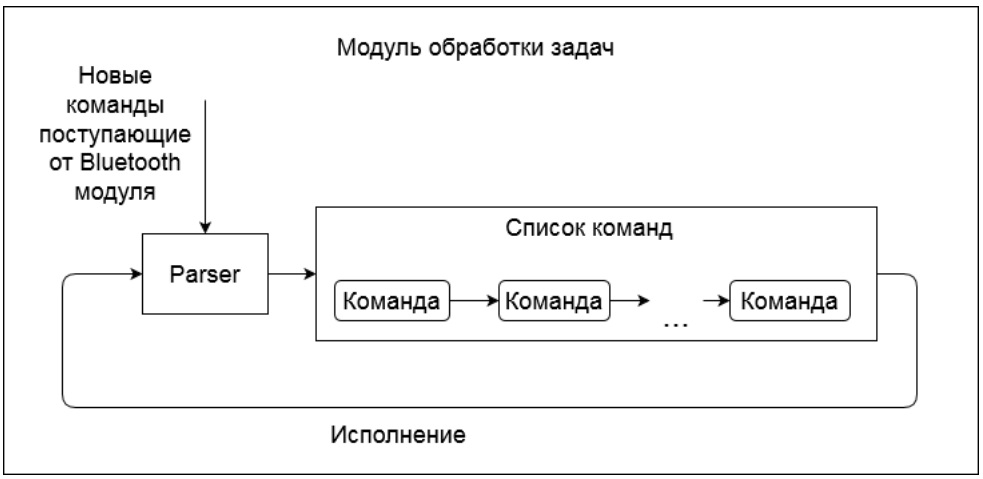

Рис. 3: Принцип работы программного обеспечения микроскопа

на рис. 3. Основу решения составляет модуль обработки задач, который выделяет время на чтение и распознавание одной пришедшей через Bluetooth команды, после чего сохраняет информацию о ней в очереди задач. Ограничение на чтение в одну команду связано с необходимостью предотвратить полную остановку исполнения остальных команд при беспрерывном приходе новых команд. После данного шага управление передается каждой из команд, присутствующих в списке задач, по очереди для их выполнения. Команды, подразумевающие длительную работу (движение приборного столика), выполняют лишь часть своей задачи, после чего передают управление следующей в списке команде. Это необходимо для поддержания псевдопараллельности выполнения задач. В случае возникновения ошибки или при завершении команды через канал связи отправляется сообщение о произошедшем событии.

3.2. Команды

Учитывая требования к программному обеспечению микроскопа были реализованы следующие команды:

- включение или выключение подсветки образца;
- передвижение приборного столика по указанной оси на определенное число шагов;
- запрос информации о текущем положении приборного столика;

- остановка движения приборного столика по указанной оси;
- остановка приборного столика по всем осям;
- калибровка положения приборного столика (считать текущее положение как нулевую координату по всем осям).

4. Протокол обмена сообщениями для мобильного приложения и микроскопа

Связь между мобильным приложением и микроскопом осуществляется с помощью Bluetooth соединения. Данное соединение гарантирует доставку пакетов с информацией и имеет контрольные суммы для их валидации. В связи с этим при разработке протокола не требовалось учитывать возможность потери данных или их повреждения. Основным требованием к разрабатываемому протоколу являлась его простота для увеличения скорости распознавания команд на стороне микроскопа. Разработанный протокол состоит из набора команд, отправляемых на микроскоп, и набора сообщений — ответов от данного прибора. Каждая отправляемая на микроскоп команда имеет уникальный идентификационный номер, позволяющий определить в ответ на какую команду микроскоп прислал то или иное сообщение. Команды, отправляемые на микроскоп.

- Команда перемещения приборного столика на определенное число шагов по указанной оси: <id>:mov <axis> <steps> <speed>$, где <id> — идентификационный номер команды, <axis> — номер оси по которой осуществлять движение, <steps> — количество шагов, на которое необходимо сместить приборный столик, <speed> — скорость передвижения приборного столика (величина задержки перед проворачиванием шагового мотора на следующих шаг в мс).

- Команда для запроса текущих координат приборного столика: <id>:get <parameter>$, где <id> — идентификационный номер команды, <parameter> — код параметра который необходимо сообщить (1 — высота, 2 — широта, 3 — долгота).

- Команда для остановки движения по указанной оси: <id>:stop <axis>$, где <id> — идентификационный номер команды, <axis> — номер оси, по которой прекратить движение.

- Команда для остановки движения по всем осям: <id>:estop$, где

<id> — идентификационный номер команды.

- Команда для калибровки (установления текущего положения приборного столика как нулевого по всем осям) микроскопа: <id>:calib$, где <id> — идентификационный номер команды.

Сообщения, получаемые от микроскопа.

- Сообщение, возвращающее запрошенную с помощью команды «get» информацию: ans <id> <value>$, где <id> — идентификационный номер команды на которую отвечает данное сообщение, <value> — значение запрашиваемого параметра.

- Сообщение о выполнении команды: ready <id>$, где <id> — идентификационный номер выполненной команды.

- Сообщение об ошибке: err <id> <error code>$, где <id> — идентификационный номер команды, при исполнении которой произошла ошибка (данный параметр может иметь значение -1, если номер команды не был установлен, например, при ошибке распознавания команды), <error code> — код ошибки.

- Сообщение об отмене выполнения команды: cancel <id> <type of canceller>$, где <id> — идентификационный номер отмененной команды, <type of canceller> — кем была отменена данная команда («u» — пользователем с помощью команд «stop» и «estop», «h» — микроскопом при попытке покинуть разрешенный интервал для перемещения приборного столика).

Для сокращения количества передаваемых данных по каналу связи все сообщения об ошибках были закодированы с помощью кода ошибки. В ходе разработки протокола был составлен следующий список возможных ошибок и их кодов:

- 1 — достигнуты границы допустимого интервала движения приборного столика;

- 2 — указан неверный код запрашиваемого значения в команде «get»;
- 3 — указано некорректное значение скорости;
- 4 — ошибка при распознавании числового значения;
- 5 — неожиданный конец команды при распознавании (указаны не все аргументы);
- 6 — указано больше аргументов для команды, чем требуется;
- 7 — ошибка при распознавании идентификационного номера команды;
- 8 — идентификационный номер команды не найден;
- 9 — команда не распознана.

5. Мобильное приложение для ОС Android

В данном разделе представлена архитектура мобильного приложения, описана реализация захвата и обработки изображений, полученных с помощью микроскопа, а также рассмотрена схема взаимодействия с данным устройством.

5.1. Архитектура

При проектировании мобильного приложения было решено использовать архитектуру Model-View-Presenter (MVP), так как она позволяет отделить код представления, описывающий внешний вид приложения и различные анимации, от бизнес-логики и упростить сохранение состояния приложения при пересоздании представления. Также данная архитектура применяется в других приложениях компании MELScience, что позволит повысить поддерживаемость разрабатываемого решения в будущем. Данная архитектура предполагает наличие трех сущностей, изображенных на рис. 4.

- Model (Модель) отвечает за бизнес-логику и содержит сущности для управления съемкой, обработкой изображений и управлением микроскопом.

- View (Представление) является тем, что видит пользователь на экране, управляет виджетами и анимациями, а также взаимодействует с пользователем, но не обрабатывает его действия самостоятельно, а передает информацию о них в представитель.

- Presenter (Представитель) отвечает за текущее состояние представления, обрабатывает переданные из представления действия пользователя и взаимодействует с моделью для получения данных. Также представитель осуществляет форматирование, а затем контроль отображения в представлении полученных данных.

Одной из главных сложностей при разработке приложений для мобильной платформы является пересоздание представления при смене

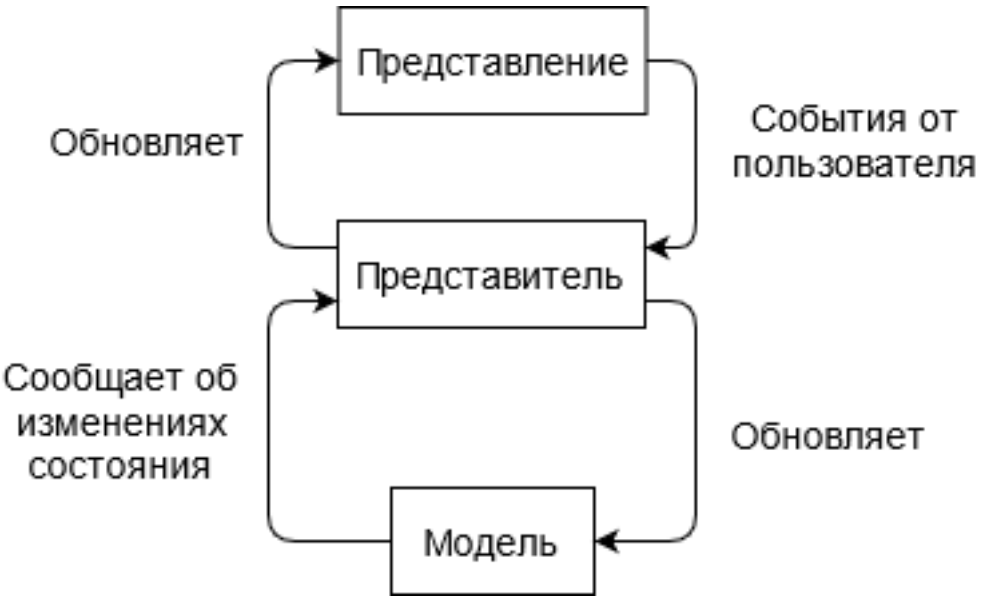

Рис. 4: Архитектура MVP

конфигурации устройства. Архитектура MVP позволяет решать данную проблему за счет хранения состояния приложения внутри представителя, который способен пережить смену конфигурации. В таком случае можно восстановить состояние представления при пересоздании, используя сохраненные в представителе данные.

Для реализации рассмотренной архитектуры была использована библиотека Moxy [22]. Данная библиотека позволяет избежать написания лишнего кода при реализации архитектуры MVP, а также поддерживает автоматическое восстановление представления после пересоздания. Это возможно засчет паттерна «команда» и специальных аннотаций, применяемых ко всем функциям, которые представитель использует для управления представлением. При выполнении данных функций, информация о них сохраняется в специальном хранилище. При пересоздании представления по данной информации восстанавливается набор ранее вызванных функций и они применяются повторно, тем самым возвращая представление в состояние предшествующее пересозданию.

5.2. Взаимодействие с микроскопом

При съемке образцов приложению требуется взаимодействовать с микроскопом и управлять движением приборного столика. Для этого необ-

ходим модуль, позволяющий устанавливать соединение с микроскопом через bluetooth, отправлять команды на передвижение приборного столика, а также получать уведомления об ошибках или успешно выполненных командах. Данный модуль был реализован, как отдельная подключаемая библиотека для возможности переиспользования в других приложениях (например, пульт для микроскопа). Архитектура разработанной библиотеки представлена на рис. 5

Основной сущностью данной библиотеки является класс Microscope, который предоставляет доступ ко всем командам, поддерживаемым на стороне микроскопа. Так как выполнение команд связано с передачей данных по сети и передвижением приборного столика микроскопа по одной или нескольким осям, что является длительными действиями относительно исполнения программного кода, то команды необходимо исполнять асинхронно. Для реализации асинхронности в библиотеке используются Kotlin Coroutines. Для формирования и распознавания сообщений согласно разработанному протоколу общения с микроскопом библиотека содержит набор реализованных классов-команд. Непосредственно связь через Bluetooth осуществляется с помощью интерфейса Connector и его реализации BluetoothConnector, взаимодействующей с модулем Bluetooth на устройстве. При необходимости можно использовать собственную реализацию данного интерфейса и тем самым легко расширить библиотеку для других каналов связи с микроскопом. Данную возможность для расширения также можно использовать при тестировании, подключив вместо реального микроскопа его эмулятор, также включенный в библиотеку.

Для упрощения работы с микроскопом в приложении были реализованы базовые программы движения, позволяющие упростить выполнение стандартных действий с микроскопом и отделить логику движения микроскопа от захвата и обработки изображений. Вместе с рассмотренной ранее библиотекой и контроллером, ответственным за инициализацию библиотеки и автоматическое подключение к прибору с использованием Bluetooth, они образуют модуль связи с микроскопом.

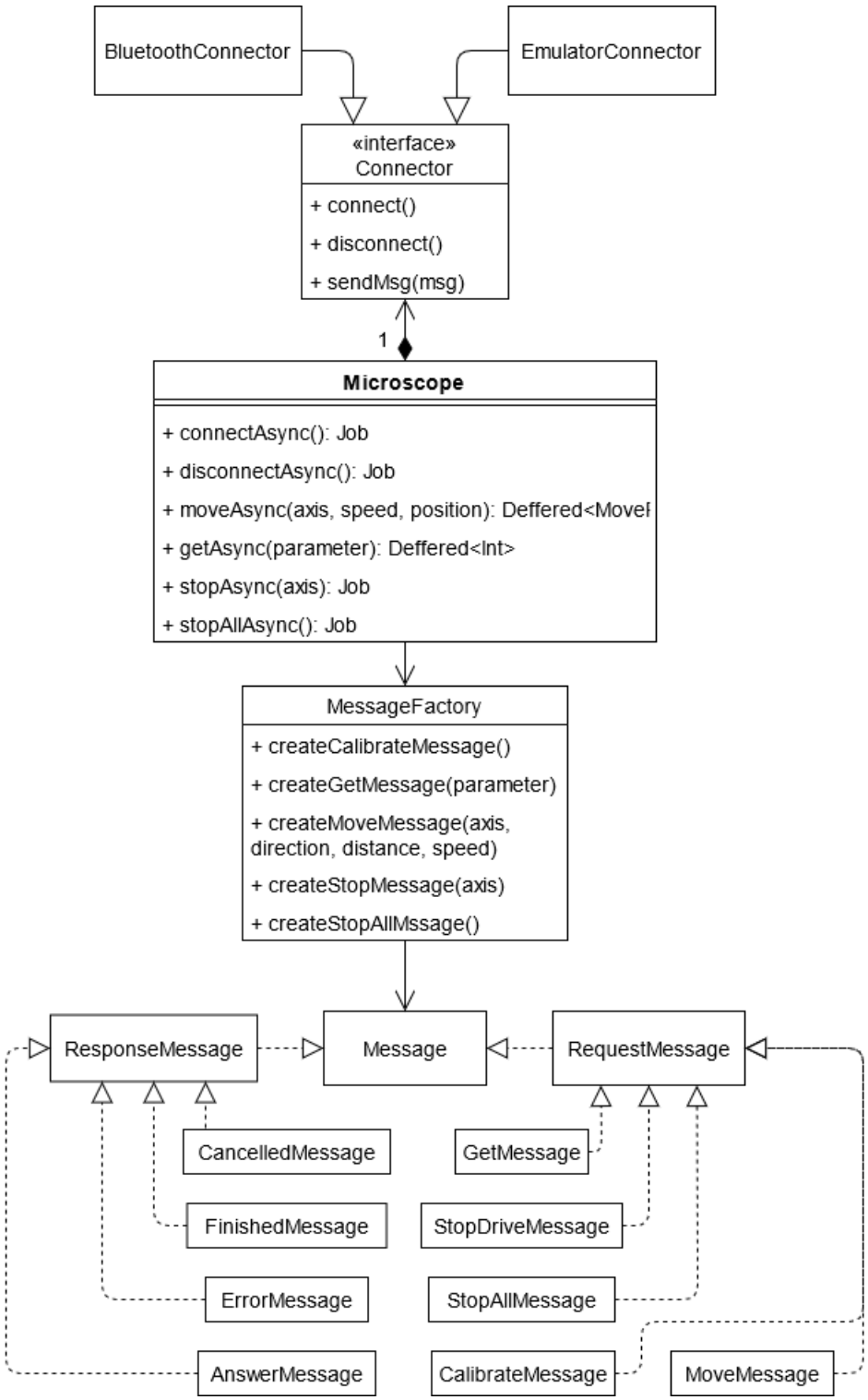

Рис. 5: Архитектура библиотеки для связи с микроскопом

5.3. Захват видеоданных

Основной задачей приложения является захват видеоданных с микроскопа и их обработка для получения полноценных изображений исследуемых образцов. Для этого необходимо получить изображение с ка-

меры смартфона, передать его в библиотеку для обработки, а затем сохранить результат. Полученный результат может быть позже обработан в встроенном фоторедакторе для использования в различных лабораторных работах. Также для получения изображения в правильной цветовой гамме и без потерь сфокусированности необходимо вручную устанавливать параметры экспозиции, автофокуса и баланса белого. Для работы с камерой было решено использовать CameraX [13]— современное api для доступа к камере в ОС Android. Данное решение было принято, так как это позволяет улучшить читаемость, а также сократить количество шаблонного кода для инициализации камеры и работы с ней. Также CameraX автоматически выполняет необходимые действия по работе с камерой при смене конфигурации приложения. Захватываемые камерой кадры необходимо не только показывать пользователю в режиме реального времени, но и передавать в мобильную библиотеку для обработки. При этом важно учитывать, что задержка для обработки кадра в мобильной библиотеке не должна влиять на захват следующих кадров. Это удалось обеспечить за счет использования встроенного в CameraX механизма, позволяющего проводить покадровую обработку с использованием различных стратегий действий в случае задержек в обработчике. В данном приложении была использована стратегия «STRATEGY_KEEP_ONLY_LATEST». Она позволяет обработчику работать необходимое ему время за счет потери части кадров. Данный подход оправдан, так как при исследовании биологических образцов скорость движения микроскопа достаточно мала.

5.4. Внедрение библиотеки для обработки изображений

Основной сущностью мобильной библиотеки, с которой должно взаимодействовать приложение, является класс VideoProcessor. Перед началом работы его необходимо инициализировать, указав какой алгоритм и с какими параметрами требуется использовать. Затем необходимо передавать в созданный экземпляр данного класса все захваты-

ваемые кадры по очереди. При этом мобильная библиотека предоставляет лишь основные алгоритмы обработки изображений, в то время как их корректное использование и комбинирование для решения соответствующей задачи является ответственностью пользователя данной библиотеки. В связи с этим для выполнения основных задач приложения был разработан подход, основанный на использовании сценариев. Каждая задача приложения (например, автофокус) представляет собой вызов данного сценария, который содержит в себе логику взаимодействия с микроскопом, а также регулирует передачу захватываемых кадров в мобильную библиотеку, самостоятельно проводя конфигурацию и инициализацию того или иного необходимого алгоритма из нее. Архитектура реализации данного подхода к обработке изображений и взаимодействию с микроскопом представлена на рис. 6

Сценарий представляет из себя набор программ движения для микроскопа, последовательно объединенных друг с другом. При этом в ходе выполнения сценария получаемые из мобильной библиотеки результаты обработки поступающих кадров используются для уточнения и параметризации следующих шагов в сценарии. В приложении были реализованы следующие сценарии.

- Автофокус. В данном сценарии основной задачей является найти максимально сфокусированное положение микроскопа и перевести в него приборный столик. Для этого используются реализованные в мобильной библиотеке объектные функции — функции, оценивающие сфокусированность кадра. В начале необходимо переместить приборный столик от самого верхнего к самому нижнему положению, вычисляя для всех захваченных кадров значение объектной функции. Затем требуется определить максимальное достигнутое значение и сопоставленное ему положение приборного столика. Заключительным шагом является перевод приборного столика в это положение.

- Ускоренный фокус-стекинг. Данный сценарий позволяет получить четкое изображение образца за время меньшее, чем при полном

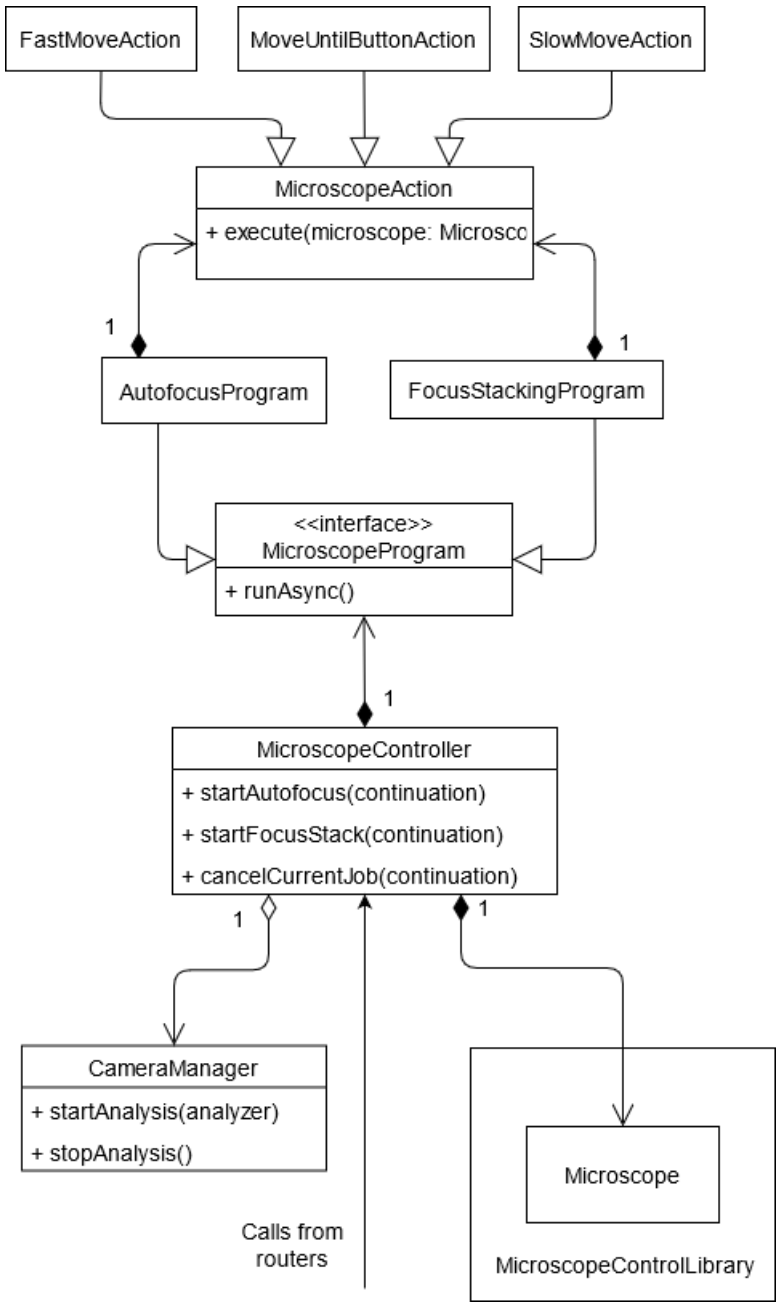

Рис. 6: Архитектура модуля сценариев работы с микроскопом

медленном проходе по всему диапазону приборного столика. Это возможно за счет быстрого предварительного прохода по всему диапазону с целью выявления области в которой препарат находится в фокусе хотя бы частично. Для выявления данной зоны используется реализованный в мобильной библиотеке алгоритм поиска пиков сфокусированности изображений. Затем обнаруженная область сканируется медленно с использованием алгоритма фокус-стекинга.

5.5. Фоторедактор

Для использования полученных с помощью приложения изображений образцов в лабораторных работах и заданиях необходимо вращать и обрезать изображения, а также отмечать и подписывать на них некоторые области. Для этого в приложении необходимо было реализовать встроенный фоторедактор, изображенный на рис. 7. При этом важно было поддержать возможность расширения функций редактора в будущем, а также функциональность по отмене ранее выполненных команд по одной.

Для этого были реализованы три виджета, отвечающие соответственно за обрезание и вращение картинки, отметки на изображении и подписи к ним. Данные виджеты способны работать как по отдельности, так и вместе со специальным классом-хранилищем, обеспечивающим возможность отмены всех использованных пользователем команд. Также это позволяет легко расширять фоторедактор путем добавления новых виджетов с соответствующей функциональностью и поддержкой нового типа команд в хранилище. Одним из примеров легкой расширяемости реализованного решения является использование для обрезания и поворота фотографий стороннего виджета uCrop [21], использующего нативную библиотеку для ускорения обработки изображений. Данный виджет был адаптирован и встроен в приложение с сохранением поддержки отмены выполненных команд.

Рис. 7: Фоторедактор

Заключение

В ходе данной работы были выполнены следующие задачи.

- Сделан обзор технологий для разработки мобильного приложения для ОС Android и используемых в приложении алгоритмов

для обработки изображений. Основываясь на изученных подходах к разработке мобильных приложений, было принято решение использовать нативный подход и разрабатывать приложение на языке Kotlin.

- Спроектировано и реализовано ПО для управления микроскопом с помощью контроллера Arduino Uno. Разработанное ПО поддерживает возможность дистанционного управления по Bluetooth и обеспечивает доступ ко всем возможностям микроскопа. Для взаимодействия с микроскопом был спроектирован протокол обмена сообщениями для мобильного приложения и микроскопа. Данный протокол ориентирован на скорость распознавания на стороне Arduino и при этом прост в чтении для человека при отладке.

- Спроектировано и реализовано мобильное приложение для ОС Android. Данное приложение способно взаимодействовать с микроскопом и выполнять цифровую обработку захваченных видеоданных. В качестве архитектуры приложения была выбрана архитектура MVP. Для возможности постобработки изображений реализован фоторедактор.

- Внедрена мобильная библиотека Владимира Кутуева для обработки изображений. Она позволяет применять алгоритмы компьютерного зрения к захватываемым с микроскопа кадрам в режиме реального времени.

Список литературы

[1] Amazon. Stepper Motor Nema 17 // официальный ритейлер. — Access mode: https://www.amazon.co.uk/dp/B00PNEQI7W/ref=as_li_ss_tl?s=electronics&keywords=nema+17&ie=UTF8&sr=1-4&linkCode=gs2&linkId=4d8077d1565e22e3079d6a4a87fd6a1d&tag=howtomuk-21 (online; accessed: 16.12.2020).

[2] Apple Inc. App Store // официальный сайт. — online; accessed: https://www.apple.com/ru/app-store/ (online; accessed: 16.12.2020).

[3] Apple Inc. iOs // официальный сайт. — online; accessed: https://www.apple.com/ru/ios (online; accessed: 16.12.2020).

[4] Arduino. Arduino // официальный сайт. — Access mode: https://www.arduino.cc/ (online; accessed: 16.12.2020).

[5] Arduino. Arduino Uno // официальный сайт. — Access mode: https://store.arduino.cc/usa/arduino-uno-rev3 (online; accessed: 16.12.2020).

[6] Automated focusing in bright-field microscopy for tuberculosis detection / O.A. OSIBOTE, R. DENDERE, S. KRISHNAN, T.S. DOUGLAS // Journal of Microscopy. — 2010. — Vol. 240, no. 2. — P. 155–163. — Access mode: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2818.2010.03389.x.

[7] Company The Qt. Qt // официальный сайт. — Access mode: https://www.qt.io (online; accessed: 16.12.2020).

[8] Complex wavelets for extended depth-of-field: A new method for the fusion of multichannel microscopy images / Brigitte Forster-Heinlein, Dimitri Van De Ville, Jesse Berent et al. // Microscopy research and technique. — 2004. — 09. — Vol. 65. — P. 33–42.

[9] Facebook Inc. Metro // официальный репозиторий. — Access mode: https://github.com/facebook/metro (online; accessed: 16.12.2020).

[10] Facebook Inc. React Native // официальный сайт. — Access mode: https://reactnative.dev (online; accessed: 16.12.2020).

[11] FocusALL: Focal Stacking of Microscopic Images Using Modified Harris Corner Response Measure / M. S. Sigdel, M. Sigdel, S. Dinç et al. // IEEE/ACM Transactions on Computational Biology and Bioinformatics. — 2016. — March. — Vol. 13, no. 2. — P. 326–340.

[12] Google. Android // официальный сайт. — online; accessed: https://www.android.com/intl/ru_ru (online; accessed: 16.12.2020).

[13] Google. CameraX // официальный сайт. — Access mode: https://developer.android.com/jetpack/androidx/releases/camera (online; accessed: 14.03.2021).

[14] Google. Flutter // официальный сайт. — Access mode: https://flutter.dev (online; accessed: 16.12.2020).

[15] Google. Google Play // официальный сайт. — online; accessed: https://play.google.com (online; accessed: 16.12.2020).

[16] MEL Science. MELScience // официальный сайт. — online; accessed: https://melscience.com/RU-ru (online; accessed: 16.12.2020).

[17] Microsoft. Microsoft // официальный сайт. — Access mode: https://www.microsoft.com (online; accessed: 16.12.2020).

[18] Microsoft. .Net // официальный сайт. — Access mode: https://dotnet.microsoft.com (online; accessed: 16.12.2020).

[19] Microsoft. Xamarin // официальный сайт. — Access mode: https://dotnet.microsoft.com/apps/xamarin (online; accessed: 16.12.2020).

[20] Tech DSD. DSD TECH Official Website // официальный сайт. — Access mode: http://www.dsdtech-global.com/2017/07/

dsd-tech-hc-05-bluetooth-serial-pass.html (online; accessed: 16.12.2020).

[21] Yalantis. uCrop // официальный репозиторий. — Access mode: https://github.com/Yalantis/uCrop (online; accessed: 14.03.2021).

[22] community Moxy. Moxy // официальный репозиторий. — Access mode: https://github.com/moxy-community/Moxy (online; accessed: 14.03.2021).

[23] Кутуев В. А. Реализация кросс-платформенной библиотеки цифровой обработки изображений в мобильной микроскопии // дипломная работа. — online; accessed: https://oops.math.spbu.ru/SE/diploma/2020/pi/Kutuev-report.pdf (online; accessed: 16.12.2020).